The University of Maryland
Department of Computer Science
A. V. Williams Building
College Park, Maryland 20742

MARUTI at ARDEC
Final Report

Submitted To:

Office of Naval Research
800 North Quincy Street
Arlington, VA 22217-5000

Sponsors ID #N000149411104
UMD ID #01-5-28734
SAI # 940209 - 7189

Principal Investigators

Ashok K. Agrawala
Satish K. Tripathi

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY (Leave blank)	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	July 11, 1996	Final Technical Report

4. TITLE AND SUBTITLE

MARUTI at ARDEC
Final Report

5. FUNDING NUMBERS

N000149411104

6. AUTHOR(S)

Ashok K. Agrawala and Satish K. Tripathi

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

University of Maryland
A. V. Williams Building
College Park, MD 20742

8. PERFORMING ORGANIZATION REPORT NUMBER

FAS 01-5-28734

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

Office of Naval Research
800 North Quincy Street
Arlington, VA 22217-5000

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION / AVAILABILITY STATEMENT

Unlimited

12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*

This is the final report of the effort undertaken at the University of Maryland, sponsored by DARPA under the DSSA (Domain Specific Software Architectures) program. The activities reported here required an active cooperation and collaboration of the US Army ARDEC (Automation and Robotics Laboratory) and the University of Maryland (Computer Science Department). The principal investigators have been developing a hard real-time operating system, MARUTI, at the University of Maryland. The goal of this effort was to demonstrate the applicability and usefulness of this operating system for exercising PID control at high frequency on the ATB1000 testbed which has been used to simulate a gun turret. A MARUTI based PID controller which operates at 200 HZ and 400 HZ was successfully implemented to demonstrate this feasibility.

14. SUBJECT TERMS

15. NUMBER OF PAGES

10

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
Unclassified	Unclassified	Unclassified	Unlimited

GENERAL INSTRUCTIONS FOR COMPLETING SF 298

The Report Documentation Page (RDP) is used in announcing and cataloging reports. It is important that this information be consistent with the rest of the report, particularly the cover and title page. Instructions for filling in each block of the form follow. It is important to ***stay within the lines*** to meet ***optical scanning requirements.***

Block 1. Agency Use Only *(Leave blank)*.

Block 2. Report Date. Full publication date including day, month, and year, if available (e.g. 1 Jan 88). Must cite at least the year.

Block 3. Type of Report and Dates Covered. State whether report is interim, final, etc. If applicable, enter inclusive report dates (e.g. 10 Jun 87 - 30 Jun 88).

Block 4. Title and Subtitle. A title is taken from the part of the report that provides the most meaningful and complete information. When a report is prepared in more than one volume, repeat the primary title, add volume number, and include subtitle for the specific volume. On classified documents enter the title classification in parentheses.

Block 5. Funding Numbers. To include contract and grant numbers; may include program element number(s), project number(s), task number(s), and work unit number(s). Use the following labels:

C	Contract	PR	Project
G	Grant	TA	Task
PE	Program Element	WU	Work Unit Accession No.

Block 6. Author(s). Name(s) of person(s) responsible for writing the report, performing the research, or credited with the content of the report. If editor or compiler, this should follow the name(s).

Block 7. Performing Organization Name(s) and Address(es). Self-explanatory.

Block 8. Performing Organization Report Number. Enter the unique alphanumeric report number(s) assigned by the organization performing the report.

Block 9. Sponsoring/Monitoring Agency Name(s) and Address(es). Self-explanatory.

Block 10. Sponsoring/Monitoring Agency Report Number. *(If known)*

Block 11. Supplementary Notes. Enter information not included elsewhere such as: Prepared in cooperation with...; Trans. of...; To be published in.... When a report is revised, include a statement whether the new report supersedes or supplements the older report.

Block 12a. Distribution/Availability Statement. Denotes public availability or limitations. Cite any availability to the public. Enter additional limitations or special markings in all capitals (e.g. NOFORN, REL, ITAR).

- DOD - See DoDD 5230.24, "Distribution Statements on Technical Documents."
- DOE - See authorities.
- NASA - See Handbook NHB 2200.2.
- NTIS - Leave blank.

Block 12b. Distribution Code.

- DOD - Leave blank.
- DOE - Enter DOE distribution categories from the Standard Distribution for Unclassified Scientific and Technical Reports.
- NASA - Leave blank.
- NTIS - Leave blank.

Block 13. Abstract. Include a brief *(Maximum 200 words)* factual summary of the most significant information contained in the report.

Block 14. Subject Terms. Keywords or phrases identifying major subjects in the report.

Block 15. Number of Pages. Enter the total number of pages.

Block 16. Price Code. Enter appropriate price code *(NTIS only)*.

Blocks 17. - 19. Security Classifications. Self-explanatory. Enter U.S. Security Classification in accordance with U.S. Security Regulations (i.e., UNCLASSIFIED). If form contains classified information, stamp classification on the top and bottom of the page.

Block 20. Limitation of Abstract. This block must be completed to assign a limitation to the abstract. Enter either UL (unlimited) or SAR (same as report). An entry in this block is necessary if the abstract is to be limited. If blank, the abstract is assumed to be unlimited.

Introduction

This is the final report of the effort undertaken at the University of Maryland, sponsored by DARPA under the DSSA (Domain Specific Software Architectures) program. The activities reported here required an active cooperation and collaboration of US Army ARDEC (Automation and Robotics Laboratory) and the University of Maryland (Computer Science Department). The principal investigators have been developing a hard real-time operating system, MARUTI, at the University of Maryland. The goal of this effort was to demonstrate the applicability and usefulness of this operating system for exercising PID control at high frequency on the ATB1000 testbed which has been used to simulate a gun turret. A MARUTI based PID controller which operates at 200 Hz and 400 Hz was sucessfully implemented to demonstrate this feasibility.

The MARUTI Project

The goal of the MARUTI project is to create an environment for the development and deployment of applications with hard real-time, fault tolerance, as well as security requirements. Such applications must be able to execute on a distributed, heterogeneous hardware base. A framework for such an environment has been created and the feasibility of the design have been demonstrated through initial implementations of the prototype components of the MARUTI environments. The development of a comprehensive environment capable of meeting the hard real-time, fault tolerance and security requirements of applications necessitates the use of new and innovative techniques for all parts of the system. Our approach continues to be that of a balance effort on both theoretical studies and prototype implementations with a continual feedback from one to the other.

The current version of MARUTI has been implemented to run in a single machine or in a multiple machine environment of PC Compatible (486) machines. Some of the tools implemented to date to support application programming include a precompiler for MPL (MARUTI Programming Language) and a processor for MCL (MARUTI Configuration Language). In addition, some scheduling analysis tools have been implemented and are ported to a 486 based platform. Detailed information about the MARUTI can be obtained from http://www.cs.umd.edu/projects/MARUTI.

The ATB1000 Testbed

The ATB1000 has been designed to assist in the development of advanced controllers for pointing turreted weapon systems accurately in harsh environments where there are many nonlinearities that degrade performance. It is an experimental fixture used for the design testing and validation of advanced control laws, as well as a test fixture for modeling turreted weapon systems to develop controllers for weapon systems without having to use the costly weapon systems themselves. The ATB1000 testbed allows the flexibility to test advanced control algorithms in a controlled test environment. A controlled performance can be tested by varying on-line nonlinearities such as backlash and friction. Once the controller passed testing on the ATB1000 testbed, it could then be tested on the actual gun systems. It was designed with nonlinearities that could be varied on-line, including two major nonlinearities: backlash and friction.

The ATB1000 was designed to simulate the types of nonlinearities and disturbances that are present in typical flexible beam stabilization problems encountered in physical systems such as a gun turret. The fixture's main body is an aluminum disc free to rotate in the horizontal plane. Affixed to the edge of the disc is an interchangeable steel rod (barrel), 1 meter in length. The control objective is to minimize the error between the commanded barrel tip position and the actual tip position.

The actuator used to control the angular motion of the disc is a brushless DC motor which transmits its torque through an on-line variable backlash mechanism and an off-line variable compliance. A mechanism is also used to introduce on-line variable friction by applying a force to the edge of the disc.

Two types of disturbances are introduced into the system. The first can be described as base motion disturbances. The base motion introduces x-y motion and rotation in the horizontal plane using four motor driven slides connected to the base through two bearings. The second disturbance attempts to simulate the effects of periodic impulsive disturbances on the system through the use of a vibrating solenoid mounted on the disc.

The beam tip position is measured using a laser-based system which is mounted on an arm which rotates in the horizontal plane. The beam tip has a flat mirror attached to it while the laser emitter and detector are attached to the arm. During experiments, that laser arm keeps the beam focused on the beam tip.

The testbed has been interfaced to a wide variety of digital controllers including the MatrixX AC100, Matlab/Dspace, and NI's LabView, and now the MARUTI operating system. The testbed was designed to include many sensors for verification of controllers as well as providing sensor inputs to adaptive controllers. The testbed has a movable base capable of 3 degree of planar motion for simulating base motion disturbances such as in helicopters, tanks, etc. Sensors include a laser tip position measuring system, accelerometers on the base itself, encoders at the control motor, after the backlash adjuster, and at the base of the inertia wheel. The beam, attached to the inertia wheel has two strain gauges mounted along the beam at 1/3 and 2/3 respectively, as well as an accelerometer and mirror at the tip. The mirror used in conjunction with the laser can supply very accurate tip position information. It should be noted that the sensors at the tip were set up for controller performance validation and model creation. Due to the harsh environment at the tip of a gun system and the high cost of specialized tip sensors it is not feasible to have such sensors available on actual gun systems.

There are two ways in which you can receive output/information from the ATB1000, one method is analog output and the other is encoder output/information. The analog output can consist of the following: quadcell position, quadcell sum, base accelerometer 1, base accelerometer 2, strain gage one, strain gage two, turret motor velocity, beam tip accelerometer, and base accelerometer 3. The encoder output can consist of the following: slide motor 1, slide motor 2, slide motor 3, slide motor 4, turret motor, backlash, laser arm, and I wheel. The analog inputs to the ATB1000 are: firing solenoid, laser arm motor, backlash setting, and friction setting.

A Schematic drawing of the ATB1000 is included on the following page (see Figure 1).

ATB1000 Testbed Structure

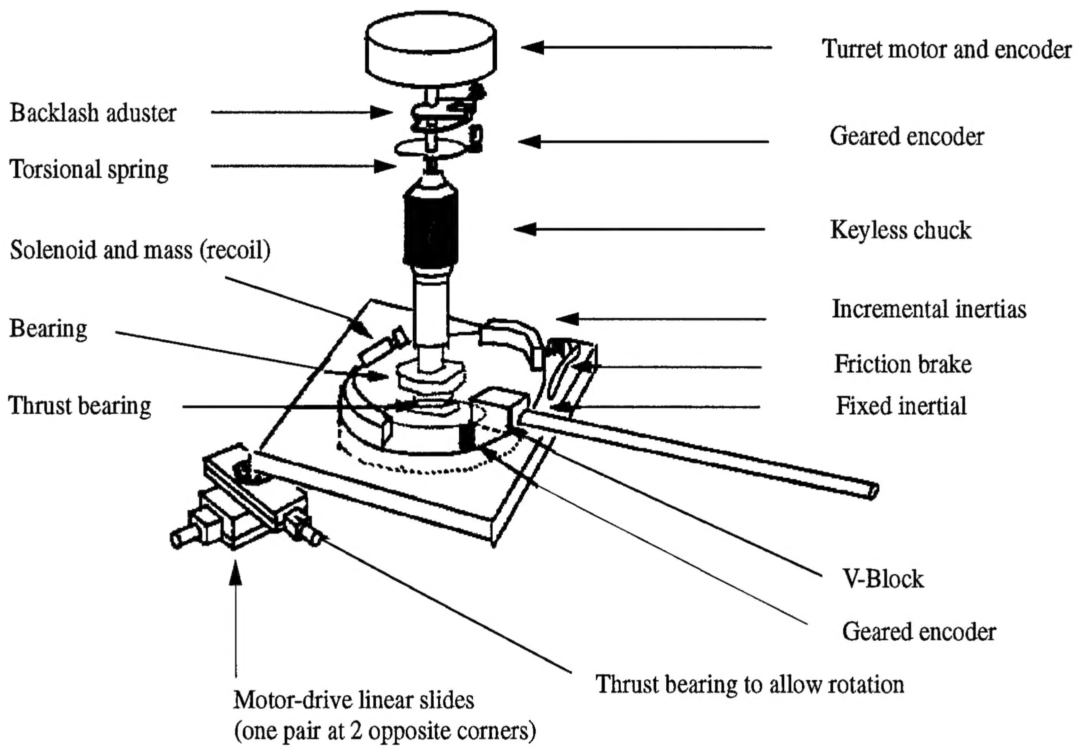

Figure 1.

PID Controller

The first PID controller consists of only the following data transfer: the encoder output is the I-Wheel and the analog input is the turret motor. Although the controller is a very simple one, the purpose of this experiment was to develop device drivers to interface with the ATB1000, develop a PID controller, interface graphics, and to validate the fact that the MARUTI operating system could be utilized to support systems that required hard real-time constraints during execution. The PID controller operates at 400 Hz and the graphics display currently operates at 50 Hz.

MARUTI Environment

Two MARUTI environments were used in this experiment: (1) development environment (2) run-time environment. The development environment ran under Berkeley Unix. It is in this environment that one needs to code in GNU C to the device drivers. In this particular case we developed device drivers for encoder boards, A/D boards, and D/A boards. We also developed in GNU C a PID controller for the I-Wheel testbed. During the developmental phase the rate at which the processes are to run is needed. The entire system (PID controller, device drivers, graphics libraries, process rates, & MARUTI libraries) was linked together so that it could be downloaded to the run-time environment. All of this was done by the system, the user merely types “mpc filename” (MARUTI Program Compile) at the command line.. The user rebooted the system under the MARUTI run-time environment and the system displayed graphical results as the system ran and tried to control the I-Wheel. Figure 2 contains the software/hardware diagram for the testbed environment, entitled “ARDEC/MARUTI Testbed Implementation”. The entire hardware development system consisted of a 486 PC, an encoder board, and an A/D D/A board.

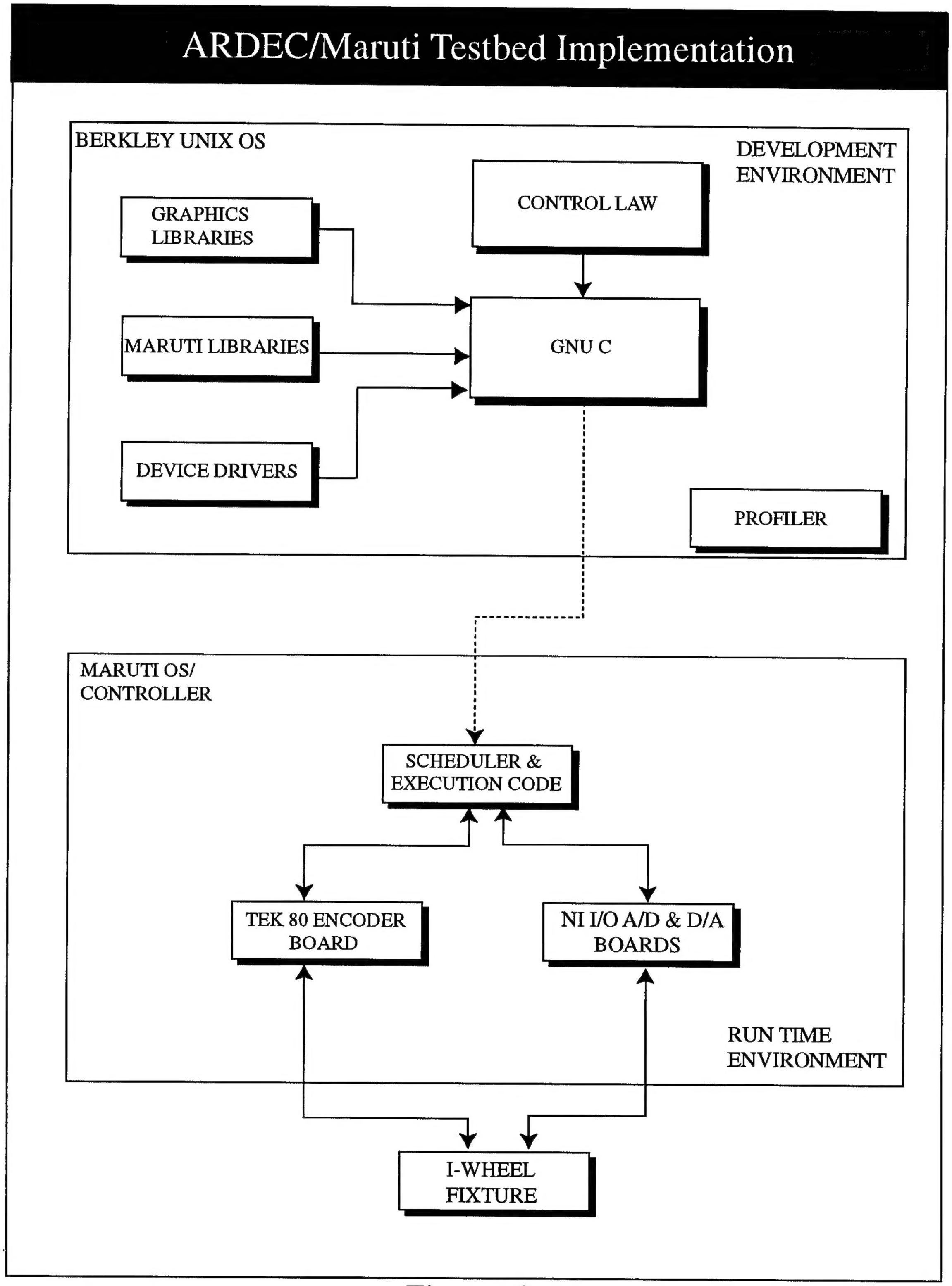
ARDEC/Maruti Testbed Implementation
BERKLEY UNIX OS
DEVELOPMENT ENVIRONMENT
GRAPHICS LIBRARIES
CONTROL LAW
MARUTI LIBRARIES
GNU C
DEVICE DRIVERS
PROFILER
MARUTI OS/ CONTROLLER
SCHEDULER & EXECUTION CODE
TEK 80 ENCODER BOARD
NI I/O A/D & D/A BOARDS
RUN TIME ENVIRONMENT
I-WHEEL FIXTURE

Figure 2.

ARDEC/MARUTI Controller Architecture

The ARDEC/MARUTI controller was developed using the "ArTek" Architecture Description Language (ADL), jointly developed by ARDEC, Automation and Robotics Lab, and Teknowledge Corporation under the ARPA "Domain Specific Software Architectures (DSSA)" program. A formal architecture description language (ADL) "schema" for describing architectures has been developed. The schema is part of a developing methodology and infrastructure for support tools to help specify, design, and validate embedded vehicle management and decision support architectures. The schema is currently in version 3.0 release and is being applied to several application programs related to embedded crew decision aids and control systems in order to test concepts methodology and tools to generate feedback from software developers. The formal language is laid out as a number of description records, each with its specific set of fields. The ADL standardizes component (defined as first class objects) interfaces which define the infrastructure for the architecture and the rescue requirements for these interconnections. The MARUTI/ARDEC PID Controller is composed of three main components: PID, Graphics, and I-Wheel. Graphically, the components are represented as ellipses, the input ports are represented as outward point on the right side of the rectangle, and output ports are represented as inward point on the left side of the rectangles. Components/elements are the basic entity within an architecture. One way to think of an architecture element is that it is a functional entity that transforms input data into output data. The connections define the topology of the architecture by defining which elements communicate with which others. When looking at the inside of the element, ports specify how the connection in the larger context relates to the implementation of the element. An example of graphical output using the ArTek architecture description language are presented in Figures 3 through 6.

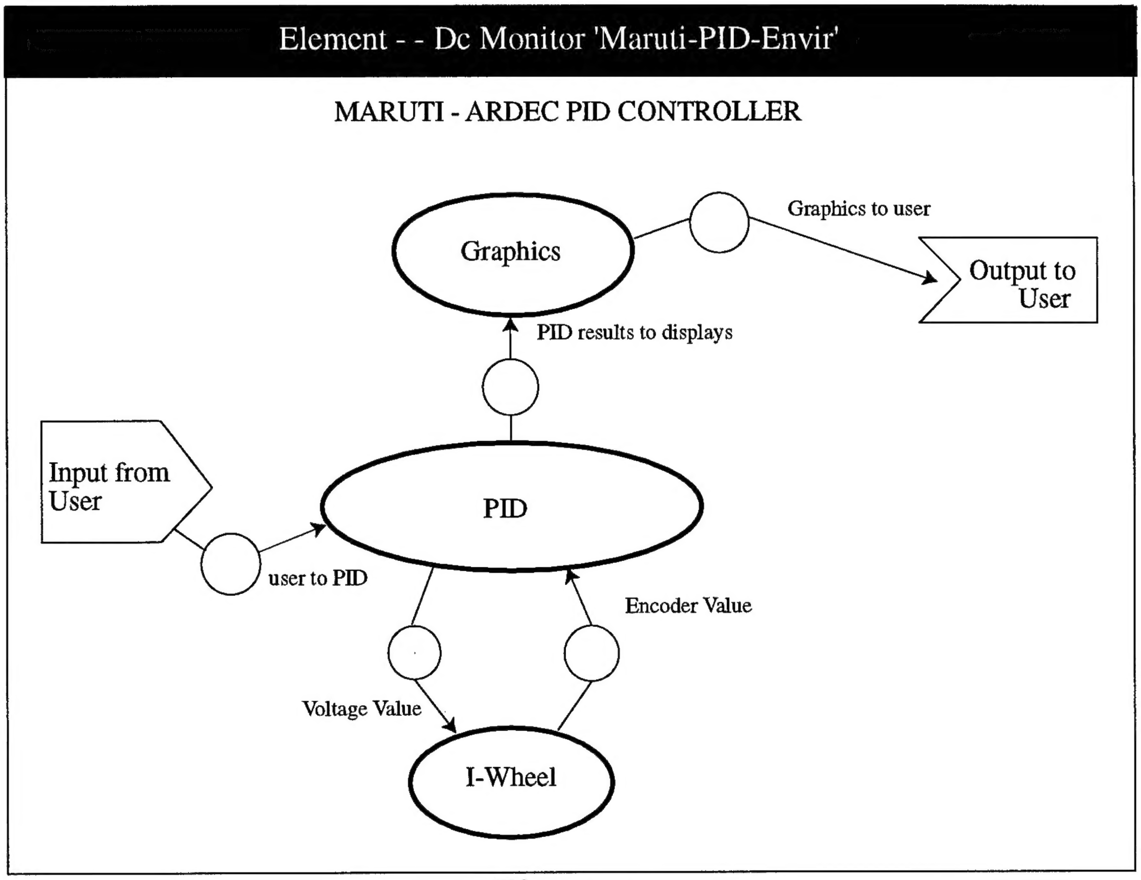

Figure 3.

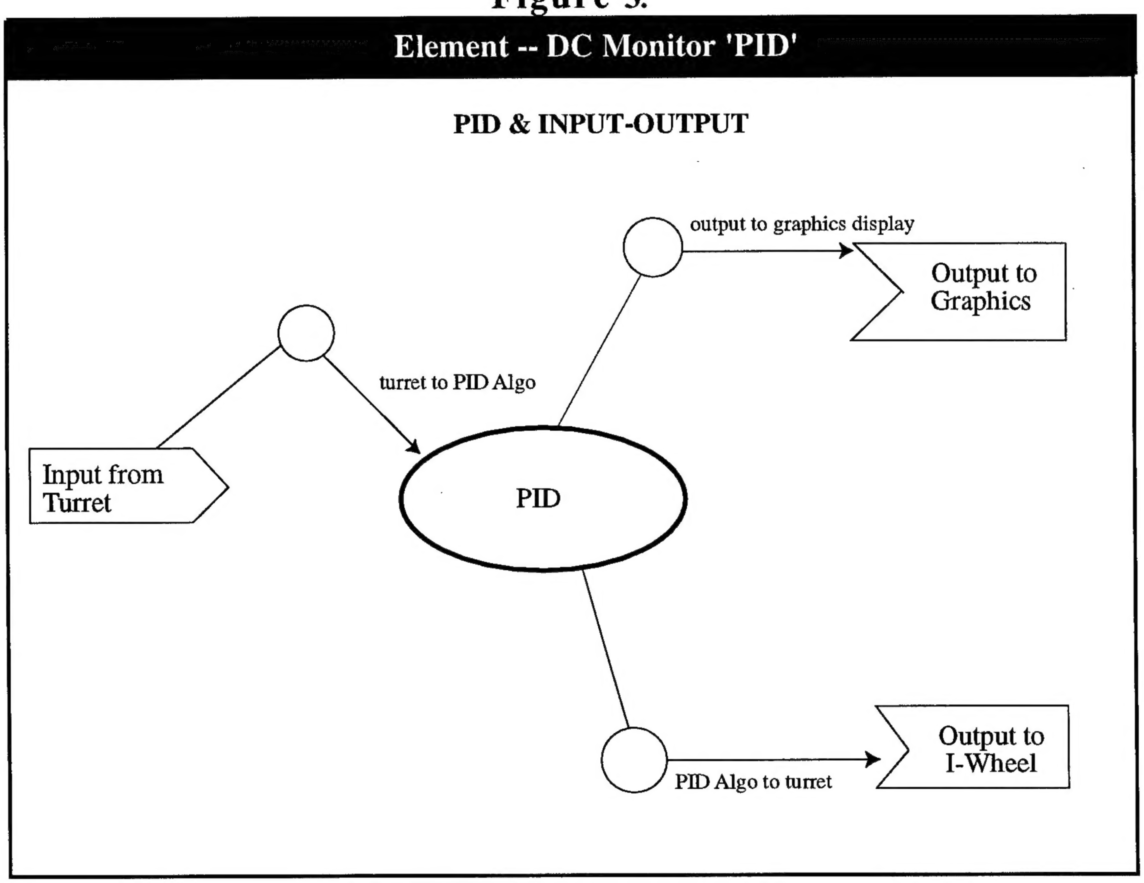

Figure 4.

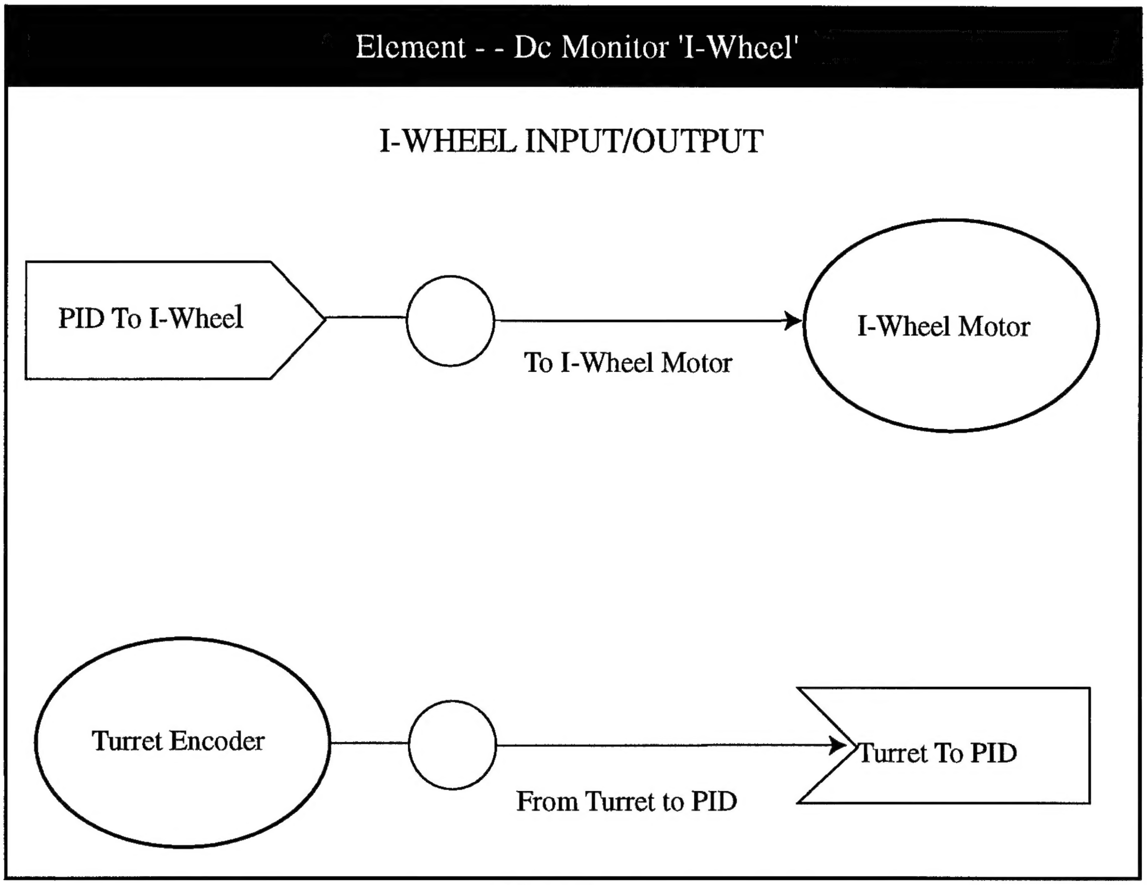

Figure 5.

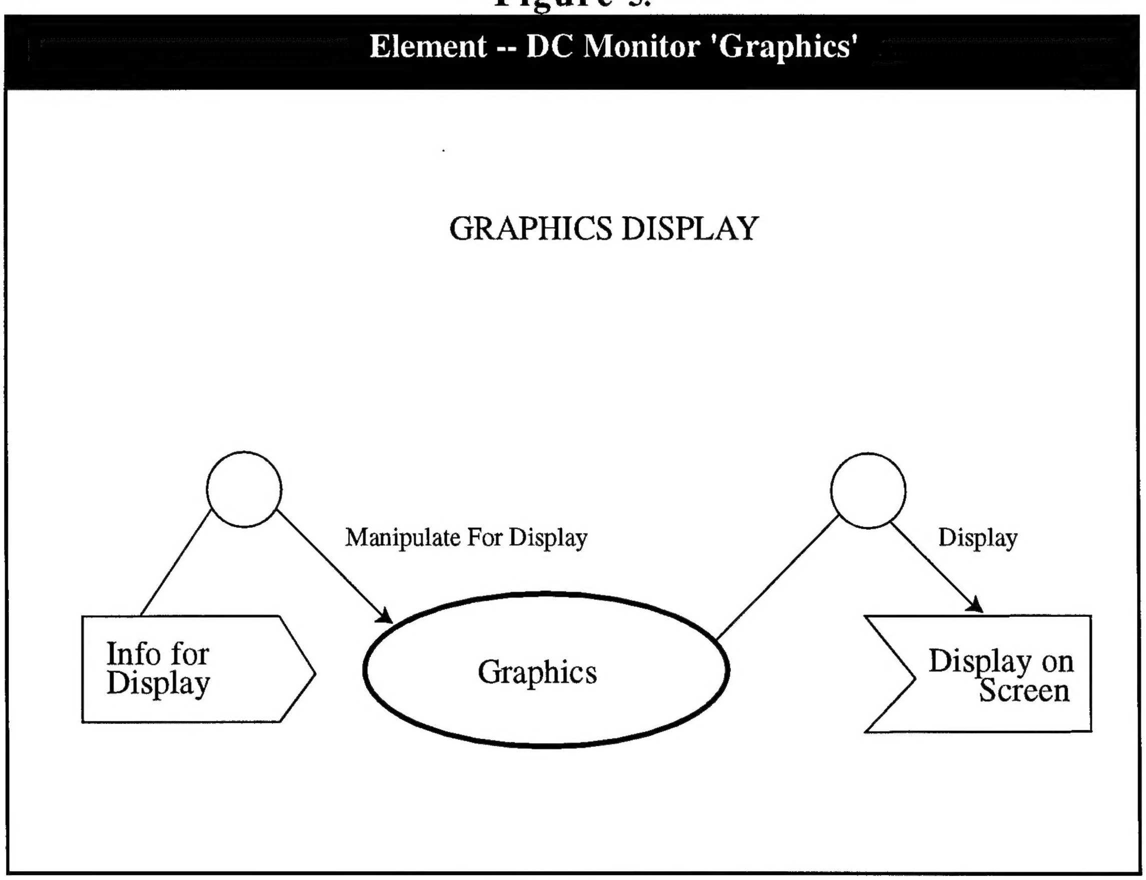

Figure 6.

Results

The PID controller inplemented using MARUTI accepted the position and velocity as inputs and produced the motor current level as the output. It was successfully demonstrated to operate at 400 Hz. It was also successfully tested to operate at 1000 Hz when the machine used was a 33Mhz 80486 processor. The software developed for this demonstration gives the user complete control of the parameters which can be changed from the keyboard at any time.

The testbed operates in two modes, tracking and positioning. It was considered necessary to operate at 200 Hz for positioning mode and 400 Hz for tracking mode. For this it was necessary that the operating system support a mode change from 400 Hz to 200 Hz. Such mode change operation was incorporated in the implementation and successfully demonstrated.

Concluding Remarks

The goal of this effort was to demonstrate the feasibility of using the MARUTI operating system for implementing the control algorithms executing at high frequencies while supporting the control functions desired by the operator. This goal was accomplished successfully.